Linear Ensembles of Word Embedding Models

Avo Muromägi
University of Tartu
Tartu, Estonia
avom@ut.ee

Kairit Sirts
University of Tartu
Tartu, Estonia
kairit.sirts@ut.ee

Sven Laur
University of Tartu
Tartu, Estonia
swen@math.ut.ee

Abstract

This paper explores linear methods for combining several word embedding models into an ensemble. We construct the combined models using an iterative method based on either ordinary least squares regression or the solution to the orthogonal Procrustes problem.

We evaluate the proposed approaches on Estonian—a morphologically complex language, for which the available corpora for training word embeddings are relatively small. We compare both combined models with each other and with the input word embedding models using synonym and analogy tests. The results show that while using the ordinary least squares regression performs poorly in our experiments, using orthogonal Procrustes to combine several word embedding models into an ensemble model leads to 7-10% relative improvements over the mean result of the initial models in synonym tests and 19-47% in analogy tests.

1 Introduction

Word embeddings—dense low-dimensional vector representations of words—have become very popular in recent years in the field of natural language processing (NLP). Various methods have been proposed to train word embeddings from unannoted text corpora (Mikolov et al., 2013b; Pennington et al., 2014; Al-Rfou et al., 2013; Turian et al., 2010; Levy and Goldberg, 2014), most well-known of them being perhaps Word2Vec (Mikolov et al., 2013b). Embedding learning systems essentially train a model from a corpus of text and the word embeddings are the model parameters. These systems contain a randomized component and so the trained models are not directly comparable, even when they have been trained on exactly the same data. This random behaviour provides an opportunity to combine several embedding models into an ensemble which, hopefully, results in a better set of word embeddings. Although model ensembles have been often used in various NLP systems to improve the overall accuracy, the idea of combining several word embedding models into an ensemble has not been explored before.

The main contribution of this paper is to show that word embeddings can benefit from ensemble learning, too. We study two methods for combining word embedding models into an ensemble. Both methods use a simple linear transformation. First of them is based on the standard ordinary least squares solution (OLS) for linear regression, the second uses the solution to the orthogonal Procrustes problem (OPP) (Schönemann, 1966), which essentially also solves the OLS but adds the orthogonality constraint that keeps the angles between vectors and their distances unchanged.

There are several reasons why using an ensemble of word embedding models could be useful. First is the typical ensemble learning argument—the ensemble simply is better because it enables to cancel out random noise of individual models and reinforce the useful patterns expressed by several input models. Secondly, word embedding systems require a lot of training data to learn reliable word representations. While there is a lot of textual data available for English, there are many smaller languages for which even obtaining enough plain unannotated text for training reliable embeddings is a problem. Thus, an ensemble approach that would enable to use the available data more effectively would be beneficial.

According to our knowledge, this is the first work that attempts to leverage the data by combining several word embedding models into a new improved model. Linear methods for combin-

Proceedings of the 21st Nordic Conference of Computational Linguistics, pages 96–104,
Gothenburg, Sweden, 23-24 May 2017.

ing two embedding models for some task-specific purpose have been used previously. Mikolov et al. (2013a) optimized the linear regression with stochastic gradient descent to learn linear transformations between the embeddings in two languages for machine translation. Mogadala and Rettinger (2016) used OPP to translate embeddings between two languages to perform cross-lingual document classification. Hamilton et al. (2016) aligned a series of embedding models with OPP to detect changes in word meanings over time. The same problem was addressed by Kulkarni et al. (2015) who aligned the embedding models using piecewise linear regression based on a set of nearest neighboring words for each word.

Recently, Yin and Schütze (2016) experimented with several methods to learn meta-embeddings by combining different word embedding sets. Our work differs from theirs in two important aspects. First, in their work each initial model is trained with a *different* word embedding system and on a *different* data set, while we propose to combine the models trained with the *same* system and on the *same* dataset, albeit using different random initialisation. Secondly, although the 1toN model proposed in (Yin and Schütze, 2016) is very similar to the linear models studied in this paper, it doesn't involve the orthogonality constraint included in the OPP method, which in our experiments, as shown later, proves to be crucial.

We conduct experiments on Estonian and construct ensembles from ten different embedding models trained with Word2Vec. We compare the initial and combined models in synonym and analogy tests and find that the ensemble embeddings combined with orthogonal Procrustes method indeed perform significantly better in both tests, leading to a relative improvement of 7-10% over the mean result of the initial models in synonym tests and 19-47% in analogy tests.

2 Combining word embedding models

A word embedding model is a matrix $W \in \mathbb{R}^{|V| \times d}$, where $|V|$ is the number of words in the model lexicon and d is the dimensionality of the vectors. Each row in the matrix W is the continuous representation of a word in a vector space.

Given r embedding models $W_1, \ldots, W_r$ we want to combine them into a target model Y. We define a linear objective function that is the sum of r linear regression optimization goals:

$$J = \sum_{i=1}^{r} \|Y - W_i P_i\|^2, \quad (1)$$

where $P_1, \ldots, P_r$ are transformation matrices that translate $W_1, \ldots, W_r$, respectively, into the common vector space containing Y.

We use an iterative algorithm to find matrices $P_1, \ldots, P_r$ and Y. During each iteration the algorithm performs two steps:

1. Solve r linear regression problems with respect to the current target model Y, which results in updated values for matrices $P_1, \ldots P_r$;
2. Update Y to be the mean of the translations of all r models:

$$Y = \frac{1}{r} \sum_{i=1}^{r} W_i P_i. \quad (2)$$

This procedure is continued until the change in the average normalised residual error, computed as

$$\frac{1}{r} \sum_{i=0}^{r} \frac{\|Y - W_i P_i\|}{\sqrt{|V| \cdot d}}, \quad (3)$$

will become smaller than a predefined threshold value.

We experiment with two different methods for computing the translation matrices $P_1, \ldots, P_r$. The first is based on the standard least squares solution to the linear regression problem, the second method is known as solution to the Orthogonal Procrustes problem (Schönemann, 1966).

2.1 Solution with the ordinary least squares (SOLS)

The analytical solution for a linear regression problem $Y = PW$ for finding the transformation matrix P, given the input data matrix W and the result Y is:

$$P = (W^T W)^{-1} W^T Y \quad (4)$$

We can use this formula to update all matrices P_i at each iteration. The problem with this approach is that because Y is also unknown and will be updated repeatedly in the second step of the iterative algorithm, the OLS might lead to solutions where both $W_i P_i$ and Y are optimized towards 0 which is not a useful solution. In order to counteract this effect we rescale Y at the start of each iteration. This is done by scaling the elements of Y so that the variance of each column of Y would be equal to 1.

	SOLS		SOPP	
Dim	**Error**	**# Iter**	**Error**	**# Iter**
50	0.162828	33	0.200994	5
100	0.168316	38	0.183933	5
150	0.169554	41	0.171266	4
200	0.172987	40	0.167554	4
250	0.175723	40	0.164493	4
300	0.177082	40	0.160988	4

Table 1: Final errors and the number of iterations until convergence for both SOLS and SOPP. The first column shows the embedding size.

.

Dim	**SOLS**	**SOPP**	**Mean W**
50	70098	**38998**	41933
100	68175	**32485**	35986
150	73182	**30249**	33564
200	73946	**29310**	32865
250	75884	**28469**	32194
300	77098	**28906**	32729
Avg	73064	**31403**	34879

Table 2: Average mean ranks of the synonym test, smaller values are better. The best result in each row is in bold. All differences are statistically significant: with $p < 2.2 \cdot 10^{-16}$ for all cases.

2.2 Solution to the Orthogonal Procrustes problem (SOPP)

Orthogonal Procrustes is a linear regression problem of transforming the input matrix W to the output matrix Y using an orthogonal transformation matrix P (Schönemann, 1966). The orthogonality constraint is specified as

$$PP^T = P^T P = I$$

The solution to the Orthogonal Procrustes can be computed analytically using singular value decomposition (SVD). First compute:

$$S = W^T Y$$

Then diagonalize using SVD:

$$S^T S = V D_S V^T$$
$$SS^T = U D_S U^T$$

Finally compute:

$$P = UV^T$$

This has to be done for each P_i during each iteration.

This approach is very similar to SOLS. The only difference is the additional orthogonality constraint that gives a potential advantage to this method as in the translated word embeddings $W_i P_i$ the lengths of the vectors and the angles between the vectors are preserved. Additionally, we no longer need to worry about the trivial solution where $P_1, \ldots, P_r$ and Y all converge towards $\mathbf{0}$.

3 Experiments

We tested both methods on a number of Word2Vec models (Mikolov et al., 2013b) trained on the Estonian Reference Corpus.[1] Estonian Reference Corpus is the largest text corpus available for Estonian. Its size is approximately 240M word tokens, which may seem like a lot but compared to for instance English Gigaword corpus, which is often used to train word embeddings for English words and which contains more than 4B words, it is quite small. All models were trained using a window size 10 and the skip-gram architecture. We experimented with models of 6 different embedding sizes: 50, 100, 150, 200, 250 and 300. For each dimensionality we had 10 models available. The number of distinct words in each model is 816757.

During training the iterative algorithm was run until the convergence threshold $th = 0.001$ was reached. The number of iterations needed for convergence for both methods and for models with different embedding size are given in Table 1. It can be seen that the convergence with SOPP took significantly fewer iterations than with SOLS. This difference is probably due to two aspects: 1) SOPP has the additional orthogonality constraint which reduces the space of feasible solutions; 2) although SOLS uses the exact analytical solutions for the least squares problem, the final solution for Y does not move directly to the direction pointed to by the analytical solutions due to the variance rescaling.

4 Results

We evaluate the goodness of the combined models using synonym and analogy tests.

4.1 Synonym ranks

One of the common ways to evaluate word embeddings is to use relatedness datasets to measure the correlation between the human and model judge-

[1] `http://www.cl.ut.ee/korpused/segakorpus`

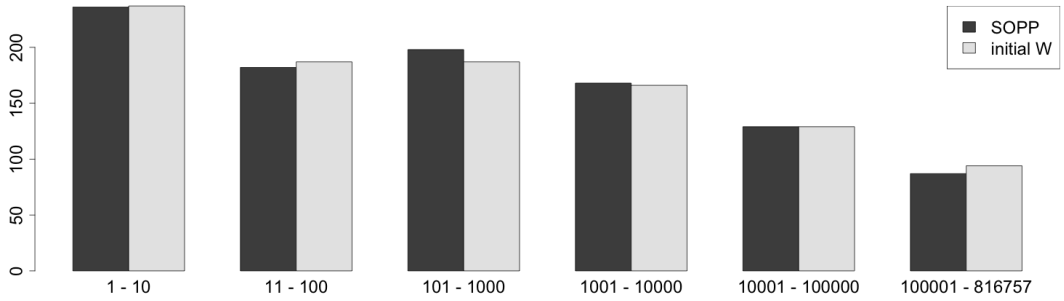

Figure 1: Histogram of the synonym ranks of the 100 dimensional vectors. Dark left columns show the rank frequencies of the SOPP model, light right columns present the rank frequencies of one of the initial models.

ments (Schnabel et al., 2015). In those datasets, there are word pairs and each pair is human annotated with a relatedness score. The evaluation is then performed by correlating the cosine similarities between word pairs with the relatedness scores. As there are no annotated relatedness datasets for Estonian, we opted to use a synonym test instead. We rely on the assumption that the relatedness between a pair of synonyms is high and thus we expect the cosine similarity between the synonymous words to be high as well.

We obtained the synonyms from the Estonian synonym dictionary.[2] We queried each word in our vocabulary and when the exact match for this word was found then we looked at the first synonym offered by the dictionary. If this synonym was present in our vocabulary then the synonym pair was stored. In this manner we obtained a total of 7579 synonym pairs. We ordered those pairs according to the frequency of the first word in the pair and chose the 1000 most frequent words with their synonyms for the synonym test.

For each first word in the synonym pair, we computed its cosine similarity with every other word in the vocabulary, ordered those similarities in the descending order and found the rank of the second word of the synonym pair in this resulting list. Then we computed the mean rank over all 1000 synonym pairs. We performed these steps on both types of combined models— Y_{SOLS} and Y_{SOPP}— and also on all input models W_i. Finally we also computed the mean of the mean ranks of all 10 input models.

The results as shown in Table 2 reveal that the synonym similarities tend to be ranked lower in the combined model obtained with SOLS when compared to the input models. SOPP, on the other hand, produces a combined model where the synonym similarities are ranked higher than in initial models. This means that the SOPP combined models pull the synonymous words closer together than they were in the initial models. The differences in mean ranks were tested using paired Wilcoxon signed-rank test at 95% confidence level and the differences were statistically significant with p-value being less than $2.2 \cdot 10^{-16}$ in all cases. In overall, the SOPP ranks are on average 10% lower than the mean ranks of the initial models. The absolute improvement on average between SOPP and mean of W is 3476.

Although we assumed that the automatically extracted synonym pairs should be ranked closely together, looking at the average mean ranks in Table 2 reveals that it is not necessarily the case— the average rank of the best-performing SOPP model is over 31K. In order to understand those results better we looked at the rank histogram of the SOPP model and one of the initial models, shown on Figure 1. Although the first bin covering the rank range from 1 to 10 contains the most words for both models and the number of synonym pairs falling to further rank bins decreases the curve is not steep and close to 100 words (87 in case of SOPP and 94 in case of the initial model) belong to the last bin counting ranks higher than 100000. Looking at the farthest synonym pairs revealed that one word in these pairs is typically polysemous and its sense in the synonym pair is a relatively rarely used sense of this word, while there are other more common senses of this word with a

[2]The Institute of the Estonian Language, `http://www.eki.ee/dict/sys/`

	Hit@1					Hit@10				
Dim	**SOLS**	**SOPP**	**Mean W**	**Min W**	**Max W**	**SOLS**	**SOPP**	**Mean W**	**Min W**	**Max W**
50	0.058	**0.193**	0.144	0.124	0.170	0.158	**0.390**	0.329	0.305	0.347
100	0.116	**0.255**	0.185	0.170	0.197	0.239	**0.475**	0.388	0.371	0.409
150	0.085	**0.278**	0.198	0.170	0.228	0.224	**0.502**	0.398	0.378	0.417
200	0.066	**0.290**	0.197	0.178	0.224	0.205	**0.541**	0.408	0.390	0.425
250	0.093	**0.282**	0.200	0.181	0.224	0.193	**0.517**	0.406	0.394	0.421
300	0.069	**0.286**	0.197	0.162	0.228	0.212	**0.533**	0.401	0.359	0.440
Avg	0.081	**0.264**	0.187			0.205	**0.493**	0.388		

Table 3: Hit@1 and Hit@10 accuracies of the analogy test. SOLS and SOPP columns show the accuracies of the combined models. Mean W, Min W and Max W show the mean, minimum and maximum accuracies of the initial models W_i, respectively. The best accuracy among the combined models and the mean of the initial models is given in bold. The last row shows the average accuracies over all embedding sizes.

completely different meaning. We give some examples of such synonym pairs:

- **kaks** (*two*) - **puudulik** (*insufficient*): the sense of this pair is the insufficient grade in high school, while the most common sense of the word **kaks** is the number *two*;
- **ida** (*east*) - **ost** (loan word from German also meaning *east*): the most common sense of the word **ost** is *purchase*;
- **rubla** (*rouble*) - **kull** (*bank note in slang*): the most common sense of the word **kull** is *hawk*.

4.2 Analogy tests

Analogy tests are another common intrinsic method for evaluating word embeddings (Mikolov et al., 2013c). A famous and typical example of an analogy question is "a man is to a king like a woman is to a ___?". The correct answer to this question is "queen".

For an analogy tuple $a : b, x : y$ (a is to b as x is to y) the following is expected in an embedding space to hold:

$$\mathbf{w}_b - \mathbf{w}_a + \mathbf{w}_x \approx \mathbf{w}_y,$$

where the vectors $\mathbf{w}$ are word embeddings. For the above example with "man", "king", "woman" and "queen" this would be computed as:

$$\mathbf{w}_{king} - \mathbf{w}_{man} + \mathbf{w}_{woman} \approx \mathbf{w}_{queen}$$

Given the vector representations for the three words in the analogy question—$\mathbf{w}_a$, $\mathbf{w}_b$ and $\mathbf{w}_x$—the goal is to maximize (Mikolov et al., 2013b)

$$cos(\mathbf{w}_y, \mathbf{w}_b - \mathbf{w}_a + \mathbf{w}_x) \quad (5)$$

over all words y in the vocabulary.

We used an Estonian analogy data set with 259 word quartets. Each quartet contains two pairs of words. The word pairs in the data set belong into three different groups where the two pairs contain either:

- a positive and a comparative adjective form, e.g. *pime* : *pimedam*, *jõukas* : *jõukam* (in English *dark* : *darker*, *wealthy* : *wealthier*);
- the nominative singular and plural forms of a noun, e.g. *vajadus* : *vajadused*, *võistlus* : *võistlused* (in English *need* : *needs* , *competition* : *competitions*);
- The lemma and the 3rd person past form of a verb, e.g. *aitama* : *aitas*, *katsuma* : *katsus* (in English *help* : *helped*, *touch* : *touched*).

We evaluate the results of the analogy test using prediction accuracy. A prediction is considered correct if and only if the vector $\mathbf{w}_y$ that maximizes (5) represents the word expected by the test case. We call this accuracy Hit@1. Hit@1 can be quite a noisy measurement as there could be several word vectors in a very close range to each other competing for the highest rank. Therefore, we also compute Hit@10, which considers the prediction correct if the word expected by the test case is among the ten closest words. As a common practice, the question words represented by the vectors $\mathbf{w}_a$, $\mathbf{w}_b$ and $\mathbf{w}_x$ were excluded from the set of possible predictions.

The Hit@1 and Hit@10 results in Table 3 show similar dynamics: combining models with SOPP is much better than SOLS in all cases. The SOPP

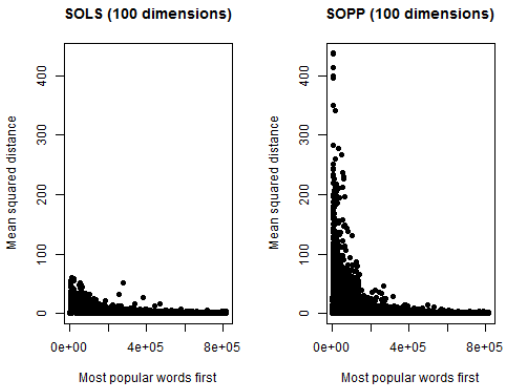

Figure 2: Mean squared distances describing the scattering of the translated word embeddings around the combined model embedding for every word in the vocabulary. The words in the horizontal axis are ordered by the frequency with most frequent words plotted first.

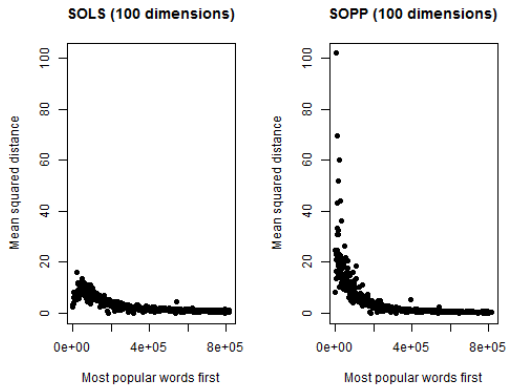

Figure 3: Mean squared distances describing the scattering of the translated word embeddings around the combined model embedding for a random sample of 1000 words. The words in the horizontal axis are ordered by the frequency with most frequent words plotted first.

combined model is better than the mean of the initial models in all six cases. Furthermore, it is consistently above the maximum of the best initial models. The average accuracy of SOPP is better than the average of the mean accuracies of initial models by 41%, relatively (7.7% in absolute) in terms of Hit@1 and 27% relatively (10.5% in absolute) in terms of Hit@10.

5 Analysis

In order to gain more understanding how the words are located in the combined model space in comparison to the initial models we performed two additional analyses. First, we computed the distribution of mean squared errors of the words to see how the translated embeddings scatter around the word embedding of the combined model. Secondly, we looked at how both of the methods affect the pairwise similarities of words.

5.1 Distribution of mean squared distances

We computed the squared Euclidean distance for each word in vocabulary between the combined model Y and all the input embedding models. The distance e_{ij} for a jth word and the ith input model is:

$$d_{ij} = \|Y_j - T_{ij}\|^2,$$

where $T_i = W_i P_i$ is the ith translated embedding model. Then we found the mean squared distance for the jth word by calculating:

$$d_j = \frac{1}{r} \sum_{i=0}^{r} d_{ij}$$

These distances are plotted on Figure 2. The words on the horizontal axis are ordered by their frequency—the most frequent words coming first. We show these results for models with 100 dimensions but the results with other embedding sizes were similar.

Notice that the distances for less frequent words are similarly small for both SOLS and SOPP methods. However, the distribution of distances for frequent words is quite different—while the distances go up with both methods, the frequent words are much more scattered when using the SOPP approach.

Figure 3 shows the mean squared distances of a random sample of 1000 words. These plots reveal another difference between the SOLS and SOPP methods. While for SOPP, the distances tend to decrease monotonically with the increase in word frequency rank, with SOLS the distances first increase and only then they start to get smaller.

Our vocabulary also includes punctuation marks and function words, which are among the most frequent tokens and which occur in many different contexts. Thus, the individual models have a lot of freedom to position them in the word embedding space. The SOLS combined model is able to bring those words more close to each other

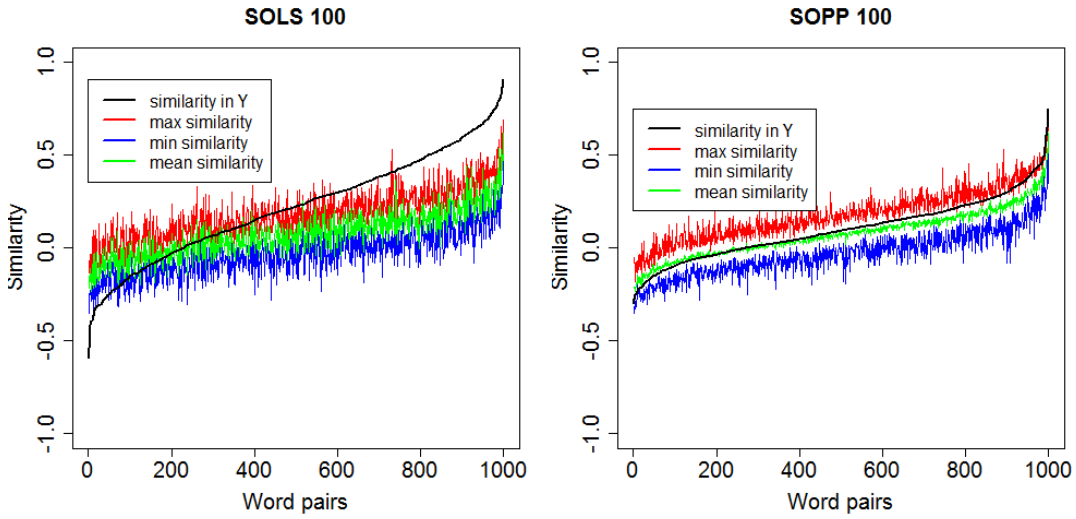

Figure 4: Cosine similarities of 1000 randomly chosen word pairs ordered by their similarity in the combined model Y. Red, blue and green bands represent the maximum, mean and minimum similarities in the initial models, respectively.

in the aligned space, while SOPP has less freedom to do that because of the orthogonality constraint. When looking at the words with largest distances under SOPP in the 1000 word random sample then we see that the word with the highest mean squared distance refers to the proper name of a well-known Estonian politician who has been probably mentioned often and in various contexts in the training corpus. Other words with a large distance in this sample include for instance a name of a month and a few quantifying modifiers.

5.2 Word pair similarities

In this analysis we looked at how the cosine similarities between pairs of words change in the combined model compared to their similarities in the input embedding models. For that, we chose a total of 1000 word pairs randomly from the vocabulary. For each pair we calculated the following values:

- cosine similarity under the combined model;
- maximum and minimum cosine similarity in the initial models W_i;
- mean cosine similarity over the initial models W_i.

The results are plotted in Figure 4. These results are obtained using the word embeddings with size 100, using different embedding sizes revealed the same patterns. In figures, the word pairs are ordered on the horizontal axis in the ascending order of their similarities in the combined model Y.

The plots reveal that 1) the words that are similar in initial models W_i are even more similar in the combined model Y; and 2) distant words in initial models become even more distant in the combined model. Although these trends are visible in cases of both SOLS and SOPP, this behaviour of the combined models to bring more similar words closer together and place less similar words farther away is more emphasized in the combined model obtained with SOLS.

In Figure 4, the red, green and blue "bands", representing the maximum, mean and minimum similarities of the initial models, respectively, are wider on the SOLS plot. This indicates that SOPP preserves more the original order of word pairs in terms of their similarities. However, some of this difference may be explained by the fact that SOPP has an overall smaller effect on the similarity compared to SOLS, which is due to the property of SOPP to preserve the angles and distances between the vectors during the transformation.

6 Discussion and future work

From the two linear methods used to combine the models, SOPP was performing consistently better in both synonym and analogy tests. Although, as shown in Figures 2 and 3, the word embeddings of the aligned initial models were more closely clus-

tered around the embeddings of the SOLS combined model, this seemingly better fit is obtained at the cost of distorting the relations between the individual word embeddings. Thus, we have provided evidence that adding the orthogonality constraint to the linear transformation objective is important to retain the quality of the translated word embeddings. This observation is relevant both in the context of producing model ensembles as well as in other contexts where translating one embedding space to another could be relevant, such as when working with semantic time series or multilingual embeddings.

In addition to combining several models trained on the same dataset with the same configuration as demonstrated in this paper, there are other possible use cases for the model ensembles which could be explored in future work. For instance, currently all our input models had the same dimensionality and the same embedding size was also used in the combined model. In future it would be interesting to experiment with combining models with different dimensionality, in this way marginalising out the embedding size hyperparameter.

Our experiments showed that the SOPP approach performs well in both synonym and analogy tests when combining the models trained on the relatively small Estonian corpus. In future we plan to conduct similar experiments on more languages that, similar to Estonian, have limited resources for training reliable word embeddings.

Another idea would be to combine embeddings trained with different models. As all word embedding systems learn slightly different embeddings, combining for instance Word2Vec (Mikolov et al., 2013b), Glove (Pennington et al., 2014) and dependency based vectors (Levy and Goldberg, 2014) could lead to a model that combines the strengths of all the input models. Yin and Schütze (2016) demonstrated that the combination of different word embeddings can be useful. However, their results showed that the model combination is less beneficial when one of the input models (Glove vectors in their example) is trained on a huge text corpus. Thus, we predict that the ensemble of word embeddings constructed based on different embedding models also has the most effect in the setting of limited training resources.

Finally, it would be interesting to explore the domain adaptation approach by combining for instance the embeddings learned from the large general domain with the embeddings trained on a smaller domain specific corpus. This could be of interest because there are many pretrained word embedding sets available for English that can be freely downloaded from the internet, while the corpora they were trained on (English Gigaword, for instance) are not freely available. The model combination approach would enable to adapt those embeddings to the domain data by making use of the pretrained models.

7 Conclusions

Although model ensembles have been often used to improve the results of various natural language processing tasks, the ensembles of word embedding models have been rarely studied so far. Our main contribution in this paper was to combine several word embedding models trained on the same dataset via linear transformation into an ensemble and demonstrate the usefulness of this approach experimentally.

We experimented with two linear methods to combine the input embedding models—the ordinary least squares solution to the linear regression problem and the orthogonal Procrustes which adds an additional orthogonality constraint to the least squares objective function. Experiments on synonym and analogy tests on Estonian showed that the combination with orthogonal Procrustes was consistently better than the ordinary least squares, meaning that preserving the distances and angles between vectors with the orthogonality constraint is crucial for model combination. Also, the orthogonal Procrustes combined model performed better than the average of the individual initial models in all synonym tests and analogy tests suggesting that combining several embedding models is a simple and useful approach for improving the quality of the word embeddings.

Acknowledgments

We thank Alexander Tkachenko for providing the pretrained input models and the analogy test questions. We also thank the anonymous reviewers for their helpful suggestions and comments.

References

Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual NLP. In *Proceedings of the Seven-*

teenth Conference on Computational Natural Language Learning, pages 183–192.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1501.

Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2015. Statistically significant detection of linguistic change. In *Proceedings of the 24th International Conference on World Wide Web*, pages 625–635.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013a. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

Tomas Mikolov, Scott Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751.

Aditya Mogadala and Achim Rettinger. 2016. Bilingual word embeddings from parallel and non-parallel corpora for cross-language text classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 692–702.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. 2015. Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 298–307.

Peter H. Schönemann. 1966. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394.

Wenpeng Yin and Hinrich Schütze. 2016. Learning Word Meta-Embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1351–1360.